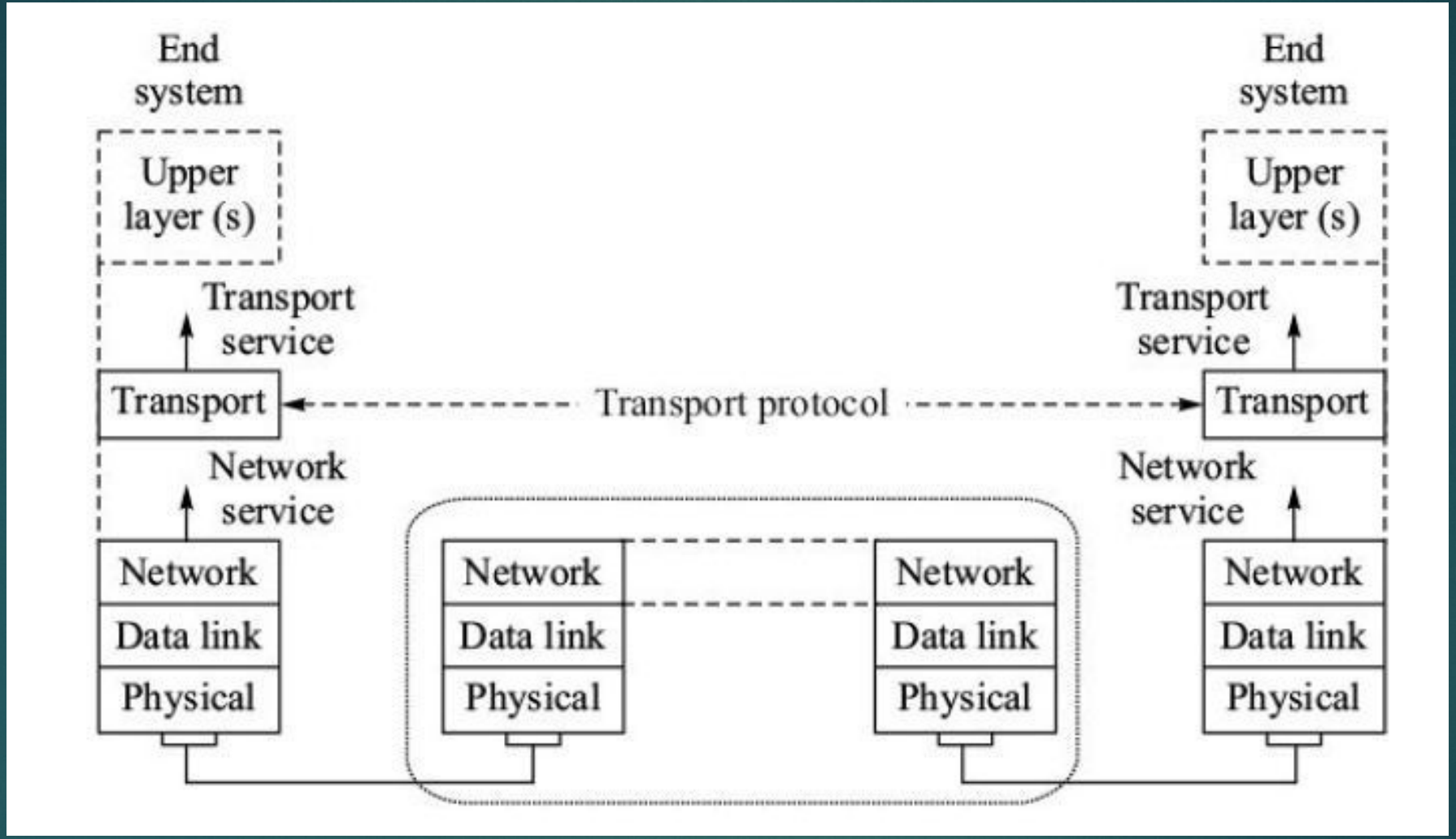
Unit-5
Transport Layer

PREPARED BY: SUSHANT BHATTARAI



Prepared By: Er. Sushant Bhattarai

Functions of Transport Layer

- Service Point Addressing
- Segmentation and reassembling
- Connection Control
- Error control
- Flow control

Prepared By: Er. Sushant Bhattarai

Client Server Paradigm

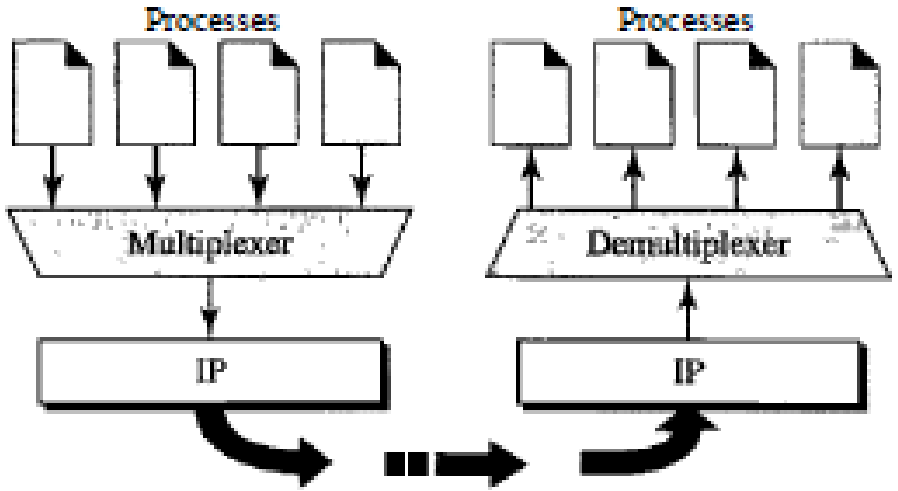
- Although there are several ways to achieve process-to-process communication, the most common one is through the client/server paradigm. A process on the local host, called a client, needs services from a process usually on the remote host, called a server.
- Both processes (client and server) have the same name. For example, to get the day and time from a remote machine, we need a Daytime client process running on the local host and a Daytime server process running on a remote machine.
- Operating systems today support both multiuser and multiprogramming environments.

Client Server Paradigm

- A remote computer can run several server programs at the same time, just as local computers can run one or more client programs at the same time. For communication, we must define the following:
- Local host
- Local process
- Remote host
- Remote process

Multiplexing and Demultiplexing

Multiplexing and demultiplexing



Multiplexing and Demultiplexing

- Multiplexing
  - At the sender site, there may be several processes that need to send packets. However, there is only one transport layer protocol at any time. This is a many-to-one relationship and requires multiplexing.
  - The protocol accepts messages from different processes, differentiated by their assigned port numbers. After adding the header, the transport layer passes the packet to the network layer.
- Demultiplexing
  - At the receiver site, the relationship is one-to-many and requires demultiplexing.
  - The transport layer receives datagrams from the network layer. After error checking and dropping of the header, the transport layer delivers each message to the appropriate process based on the port number

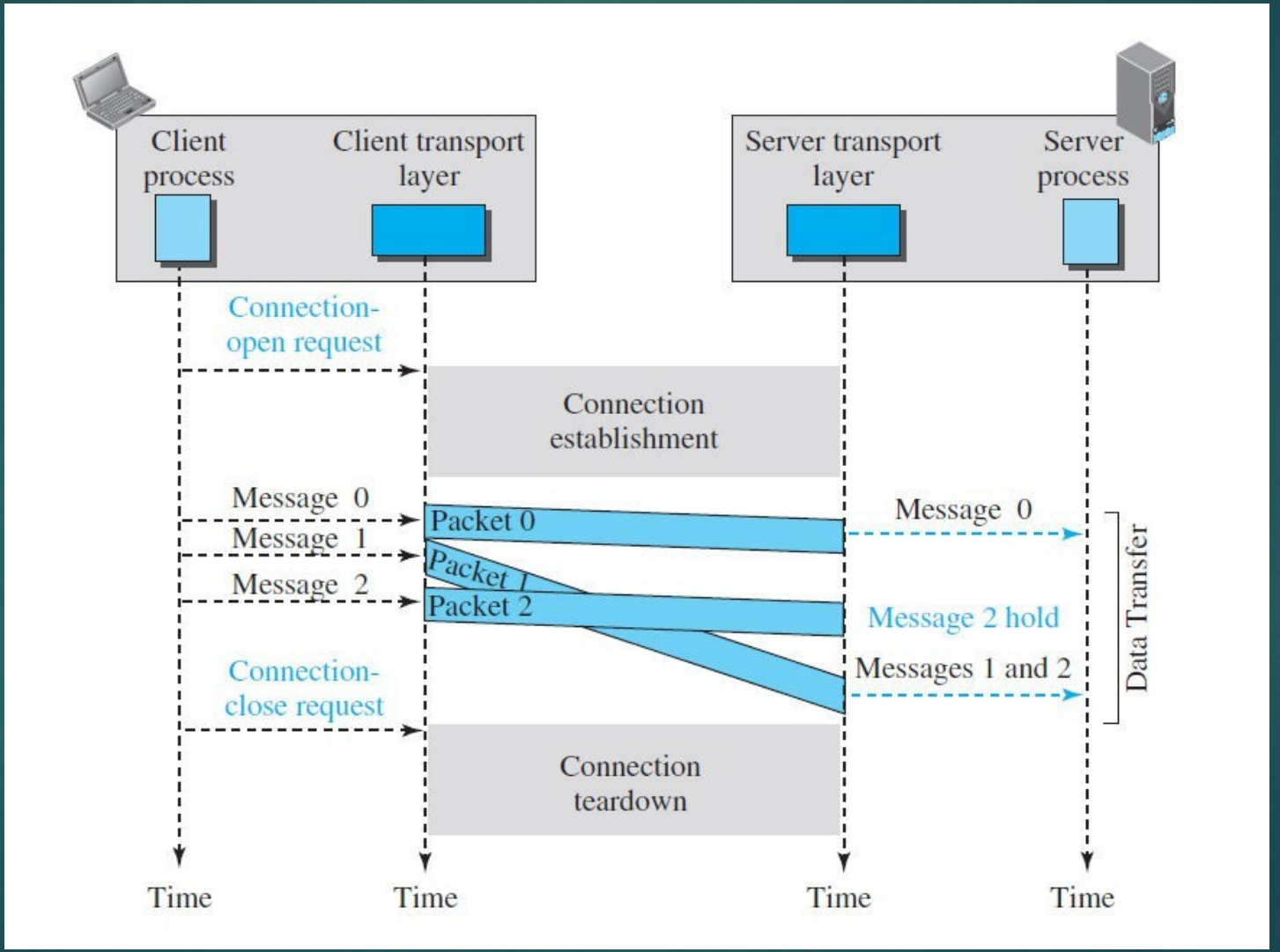
Connection oriented Services

- In a connection-oriented service, the client and the server first need to establish a logical connection between themselves.
- The data exchange can only happen after the connection establishment.

After data exchange, the connection needs to be torn down

- the connection-oriented service at the transport layer is different from the same service at the network layer.
- In the network layer, connection oriented service means a coordination between the two end hosts and all the routers in between.
- At the transport layer, connection-oriented service involves only the two hosts; the service is end to end.

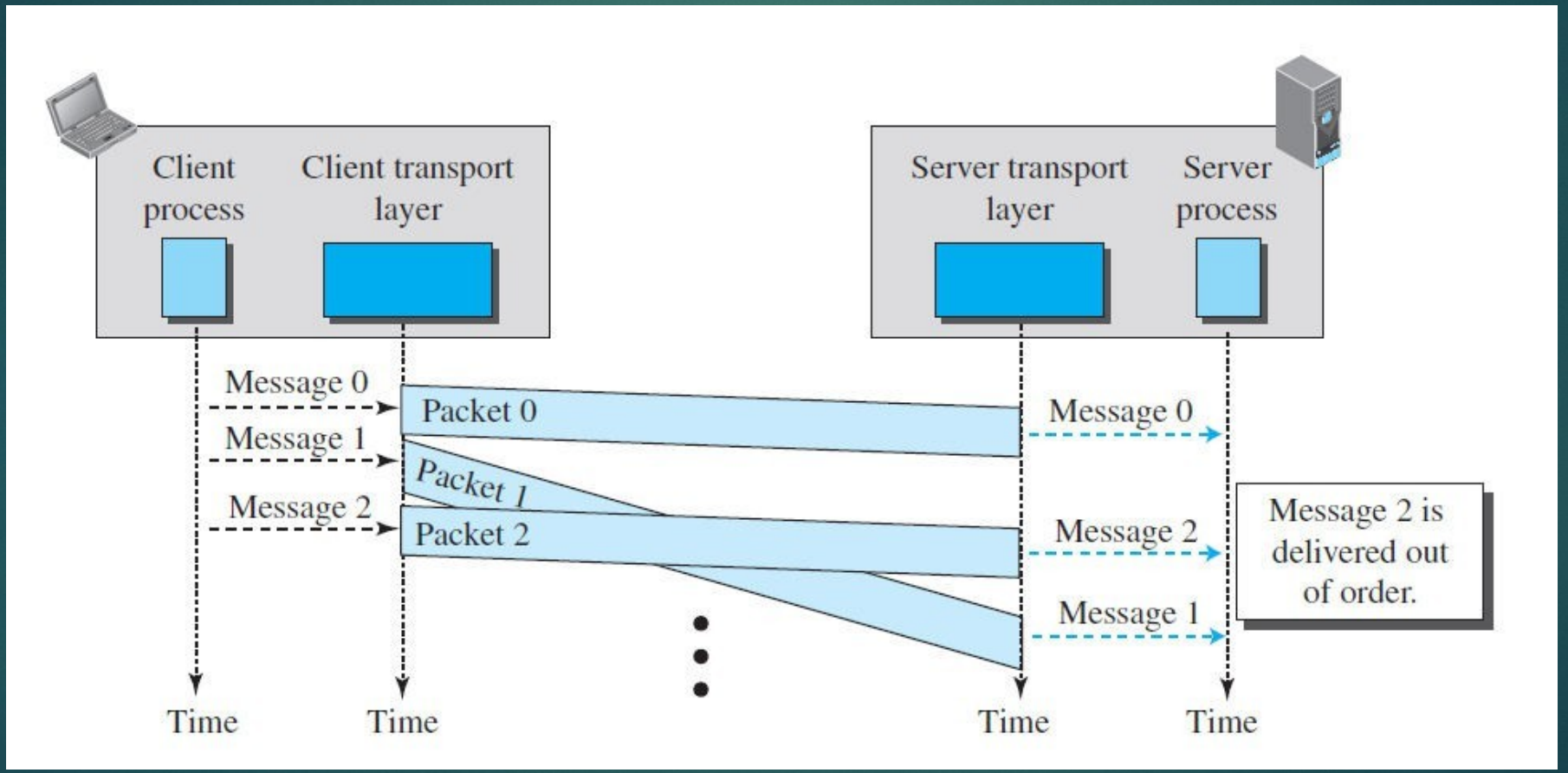
Connection oriented service



Connectionless service

- In a connectionless service, the source process (application program) needs to divide its message into chunks of data of the size acceptable by the transport layer and deliver them to the transport layer one by one.
- The transport layer treats each chunk as a single unit without any relation between the chunks. When a chunk arrives from the application layer, the transport layer encapsulates it in a packet and sends it.
- To show the independency of packets, assume that a client process has three chunks of messages to send to a server process. The chunks are handed over to the connectionless transport protocol in order.
- However, since there is no dependency between the packets at the transport layer, the packets may arrive out of order at the destination and will be delivered out of order to the server process

Connectionless service



Connectionless service

- In Figure, we have shown the movement of packets using a time line, but we have assumed that the delivery of the process to the transport layer and vice versa are instantaneous.
- The figure shows that at the client site, the three chunks of messages are delivered to the client transport layer in order (0, 1, and 2). Because of the extra delay in transportation of the second packet, the delivery of messages at the server is not in order (0, 2, 1).
- If these three chunks of data belong to the same message, the server process may have received a strange message.

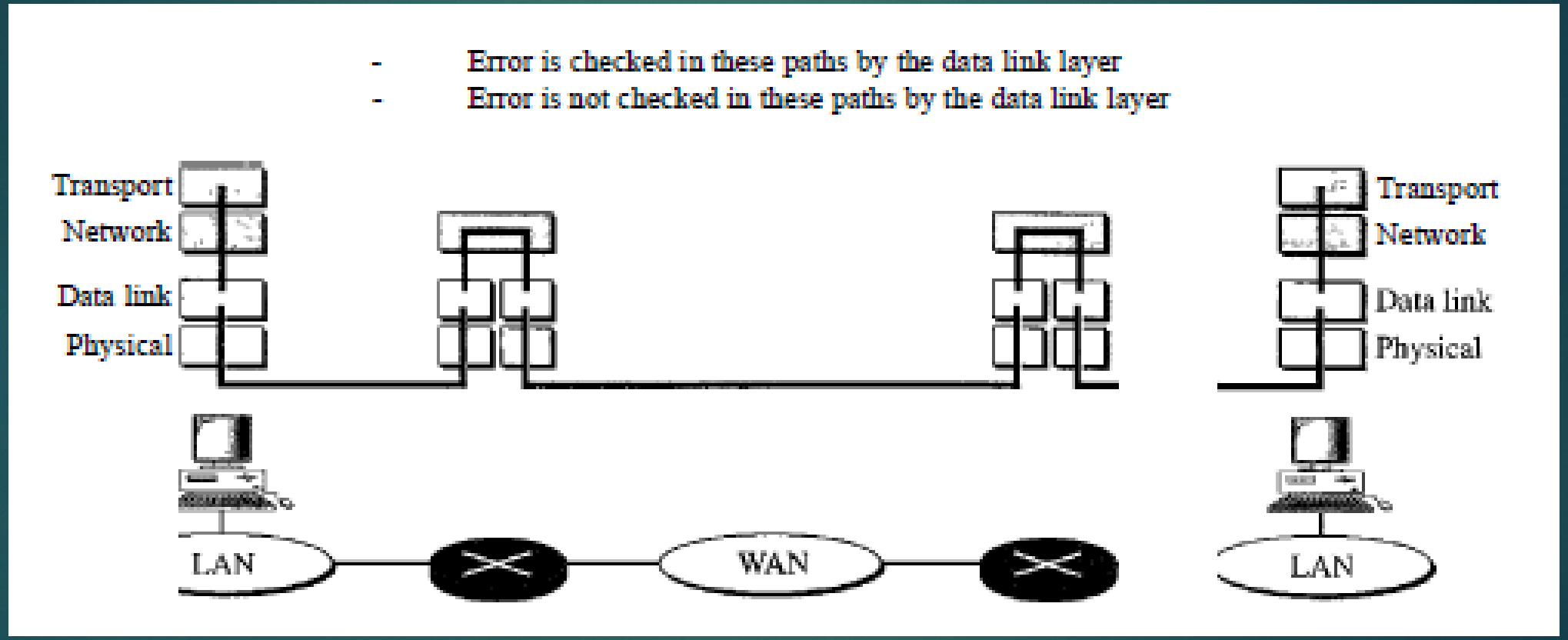
Reliable VS Unreliable

- The transport layer service can be reliable or unreliable. If the application layer program needs reliability, we use a reliable transport layer protocol by implementing flow and error control at the transport layer.
- This means a slower and more complex service. On the other hand, if the application program does not need reliability because it uses its own flow and error control mechanism or it needs fast service or the nature of the service does not demand flow and error control (real-time applications), then an unreliable protocol can be used.
- In the Internet, there are three common different transport layer protocols, as we have already mentioned. UDP is connectionless and unreliable; TCP and SCTP are connection oriented and reliable. These three can respond to the demands of the application layer programs.

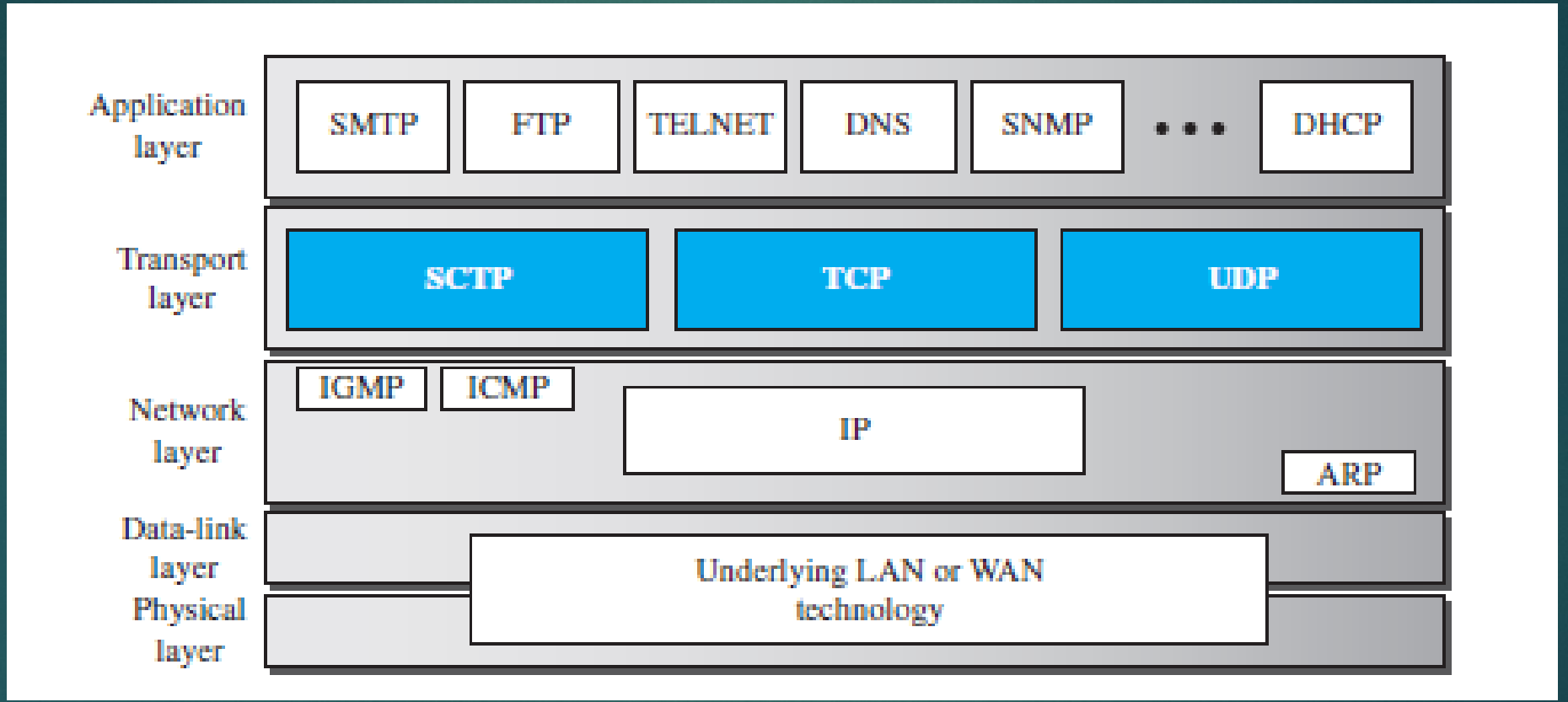
Reliable VS Unreliable

- One question often comes to the mind. If the data link layer is reliable and has flow and error control, do we need this at the transport layer, too? The answer is yes.
- Reliability at the data link layer is between two nodes; we need reliability between two ends. Because the network layer in the Internet is unreliable (best-effort delivery), we need to implement reliability at the transport layer.
- To understand that error control at the data link layer does not guarantee error control at the transport layer, let us look at Figure below.

Reliable VS Unreliable



Transport Protocol



Prepared By: Er. Sushant Bhattarai

Services

- Each protocol provides a different type of service and should be used appropriately.
- **UDP**
  - UDP is an unreliable connectionless transport-layer protocol used for its simplicity and efficiency in applications where error control can be provided by the application-layer process.
- **TCP**
  - TCP is a reliable connection-oriented protocol that can be used in any application where reliability is important.
- **SCTP**
  - SCTP is a new transport-layer protocol that combines the features of UDP and TCP.

Prepared By: Er. Sushant Bhattarai

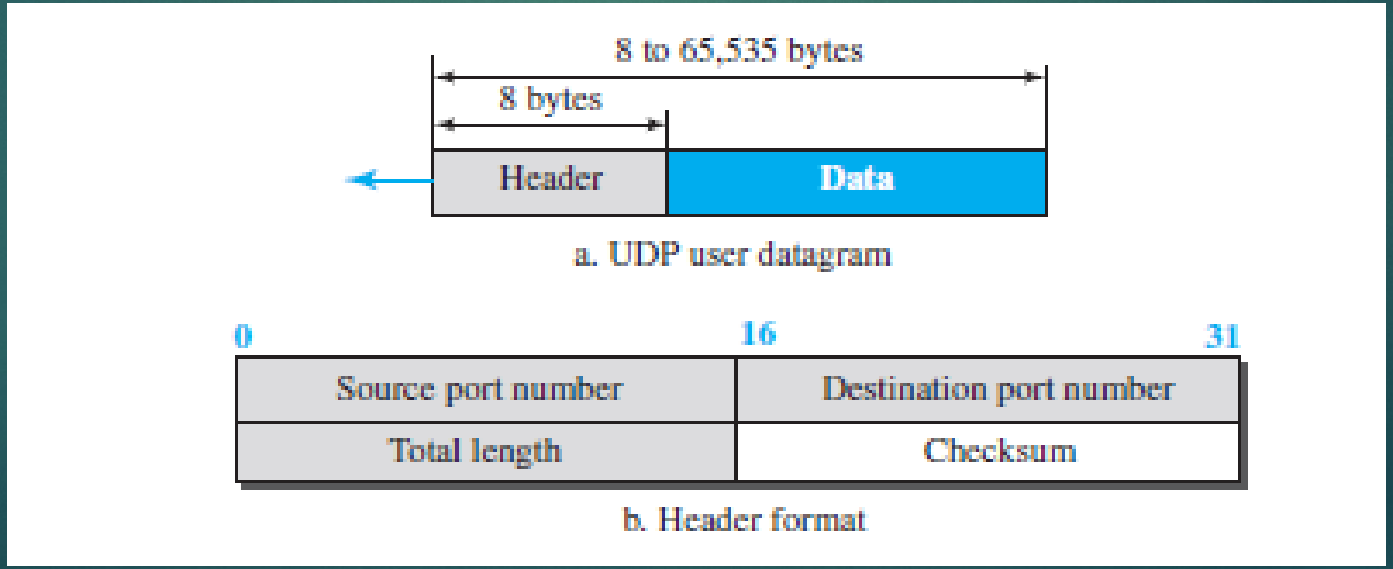
UDP

- User Datagram Protocol (UDP) is a connectionless, unreliable transport protocol.
- It does not add anything to the services of IP except for providing process- to-process communication
- UDP is a very simple protocol using a minimum of overhead.
- If a process wants to send a small message and does not care much about reliability, it can use UDP.
- Sending a small message using UDP takes much less interaction between the sender and receiver than using TCP.

Prepared By: Er. Sushant Bhattarai

UDP datagram

- UDP packets, called user datagrams, have a fixed-size header of 8 bytes made of four fields, each of 2 bytes (16 bits).
- The first two fields define the source and destination port numbers. The third field defines the total length of the user datagram, header plus data. The last field can carry the optional checksum



Prepared By: Er. Sushant Bhattarai

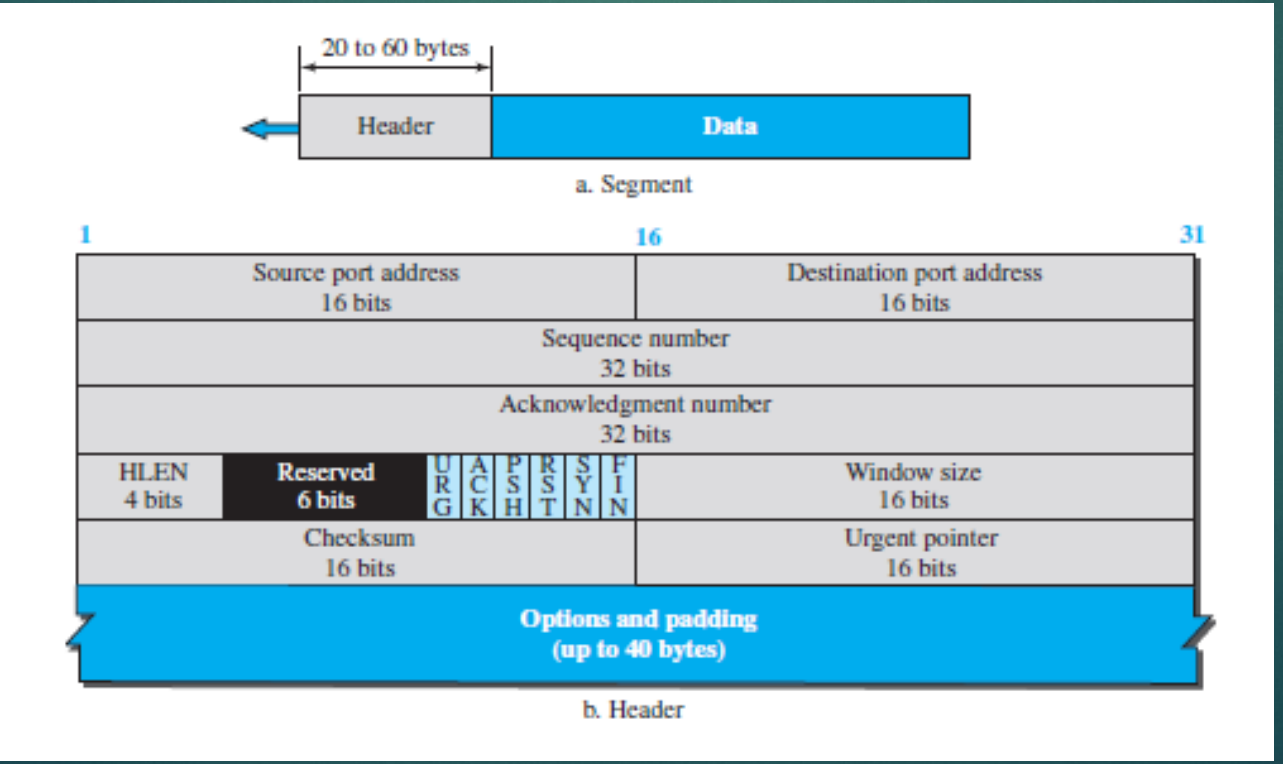
TCP

- Transmission Control Protocol (TCP) is a connection-oriented, reliable protocol.
- TCP explicitly defines connection establishment, data transfer, and connection teardown phases to provide a connection-oriented service.
- TCP uses a combination of GBN and SR protocols to provide reliability.
- To achieve this goal, TCP uses checksum (for error detection), retransmission of lost or corrupted packets, cumulative and selective acknowledgments, and timers

Prepared By: Er. Sushant Bhattarai

TCP segment

- The segment consists of a header of 20 to 60 bytes, followed by data from the application program.
- The header is 20 bytes if there are no options and up to 60 bytes if it contains options.



Prepared By: Er. Sushant Bhattarai

TCP segment

- Source port address. This is a 16-bit field that defines the port number of the application program in the host that is sending the segment.
- Destination port address. This is a 16-bit field that defines the port number of the application program in the host that is receiving the segment.
- Sequence number. This 32-bit field defines the number assigned to the first byte of data contained in this segment. To ensure connectivity, each byte to be transmitted is numbered. The sequence number tells the destination which byte in this sequence is the first byte in the segment.
- Acknowledgment number. This 32-bit field defines the byte number that the receiver of the segment is expecting to receive from the other party. If the receiver of the segment has successfully received byte number x from the other party, it returns x + 1 as the acknowledgment number.

TCP Segment

- Header length. This 4-bit field indicates the number of 4-byte words in the TCP header. The length of the header can be between 20 and 60 bytes. Therefore, the value of this field is always between 5 (5 × 4 = 20) and 15 (15
× 4 = 60).
- Control. This field defines 6 different control bits or flags, One or more of these bits can be set at a time. These bits enable flow control, connection establishment and termination, connection abortion, and the mode of data transfer in TCP.

URG: Urgent pointer is valid
ACK: Acknowledgment is valid
PSH : Request for push
RST : Reset the connection
SYN: Synchronize sequence numbers
FIN : Terminate the connection

TCP segment

- Window size. This field defines the window size of the sending TCP in bytes. Note that the length of this field is 16 bits, which means that the maximum size of the window is 65,535 bytes. This value is normally referred to as the receiving window (rwnd) and is determined by the receiver. The sender must obey the dictation of the receiver in this case.
- Checksum. This 16-bit field contains the checksum. The calculation of the checksum for TCP follows the same procedure as the one described for UDP. However, the use of the checksum in the UDP datagram is optional, whereas the use of the checksum for TCP is mandatory.
- Urgent pointer. This 16-bit field, which is valid only if the urgent flag is set, is used when the segment contains urgent data. It defines a value that must be added to the sequence number to obtain the number of the last urgent byte in the data section of the segment.
- Options. There can be up to 40 bytes of optional information in the TCP header.

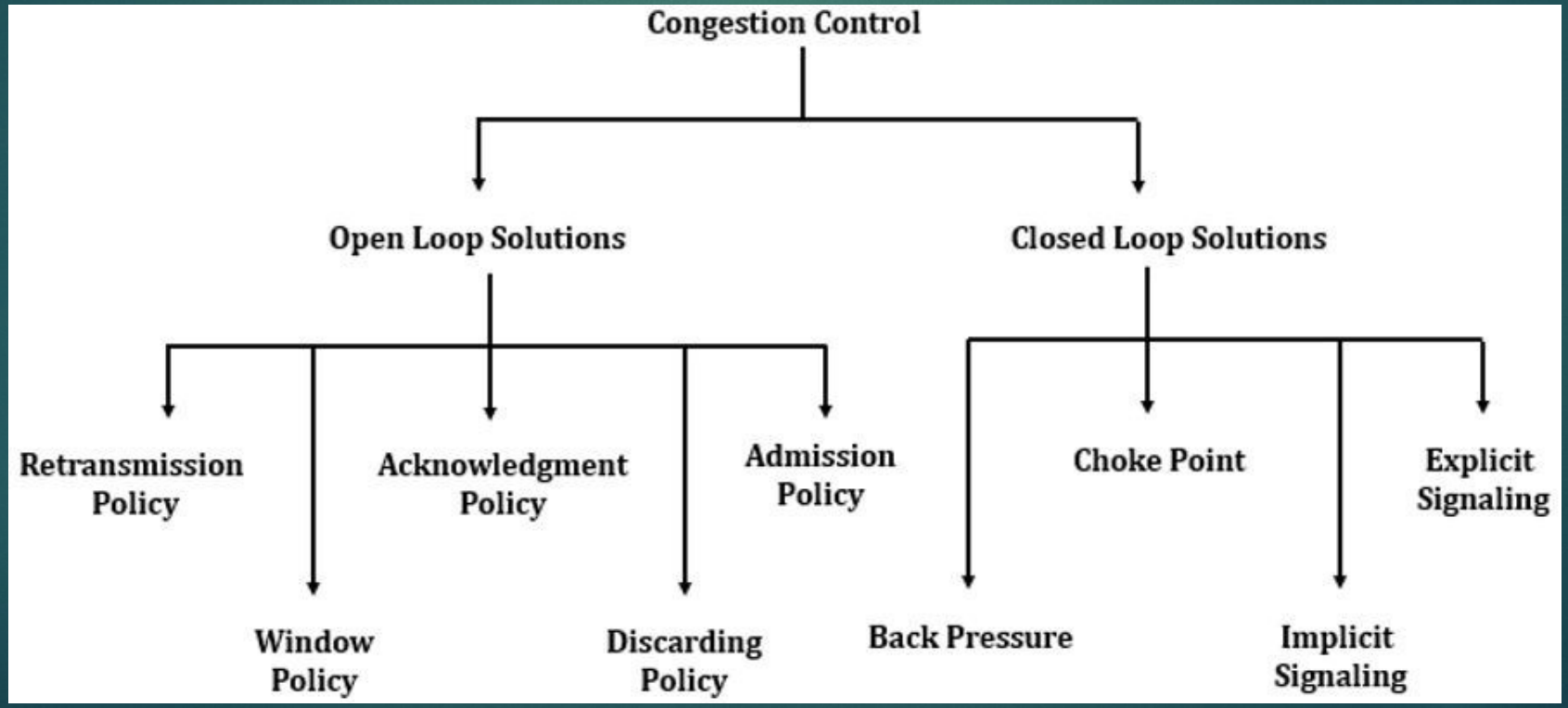
Prepared By: Er. Sushant Bhattarai

Protocol	TCP	UDP
Connection	connection-oriented	connectionless
Usage	high reliability, critical-less trans-mission time	fast, efficient transm-ission, small queries, huge numbers of clients
Ordering of data packets	rearranges packets in order	no inherent order
Reliability	yes	no
Streaming of data	read as a byte stream	sent and read indivi-dually
Error checking	error checking and recovery	simply error checking, no error recovery
Acknowle-dgement	acknowledgement segments	no acknowledgment

Congestion Control

Prepared By: Er. Sushant Bhattarai

- Congestion Control refers to techniques and mechanisms that can either prevent congestion, before it happens, or remove congestion, after it has happened.



Open Loop congestion control

- In open-loop congestion control, policies are applied to prevent congestion before it happens. In these mechanisms, congestion control is handled by either the source or the destination.
- Retransmission Policy
- Window Policy
- Acknowledgment Policy
- Discarding Policy
- Admission Policy

Prepared By: Er. Sushant Bhattarai

Open Loop Congestion Control

- Retransmission Policy
  - Retransmission is sometimes unavoidable.
  - If the sender feels that a sent packet is lost or corrupted, the packet needs to be retransmitted. Retransmission in general may increase congestion in the network. However, a good retransmission policy can prevent congestion.
  - The retransmission policy and the retransmission timers must be designed to optimize efficiency and at the same time prevent congestion.
- Window Policy
  - The type of window at the sender may also affect congestion.
  - The Selective Repeat window is better than the Go-Back-*N* window for congestion control. In the Go-Back-*N* window, when the timer for a packet times out, several packets may be resent, although some may have arrived safe and sound at the receiver.
  - This duplication may make the congestion worse. The Selective Repeat window, on the other hand, tries to send the specific packets that have been lost or corrupted

Prepared By: Er. Sushant Bhattarai

Open Loop Congestion Control

Prepared By: Er. Sushant Bhattarai

- Acknowledgment Policy
  - The acknowledgment policy imposed by the receiver may also affect congestion. If the receiver does not acknowledge every packet it receives, it may slow down the sender and help prevent congestion.
  - Several approaches are used in this case. A receiver may send an acknowledgment only if it has a packet to be sent or a special timer expires. A receiver may decide to acknowledge only *N* packets at a time.
- Discarding Policy
  - A good discarding policy by the routers may prevent congestion and at the same time may not harm the integrity of the transmission.
  - For example, in audio transmission, if the policy is to discard less sensitive packets when congestion is likely to happen, the quality of sound is still preserved and congestion is prevented or alleviated.

Open Loop Congestion Control

- Admission policy
  - An admission policy, can also prevent congestion in virtual-circuit networks. Switches in a flow first check the resource requirement of a flow before admitting it to the network.
  - A router can deny establishing a virtual-circuit connection if there is congestion in the network or if there is a possibility of future congestion

Prepared By: Er. Sushant Bhattarai

Closed Loop Congestion Control

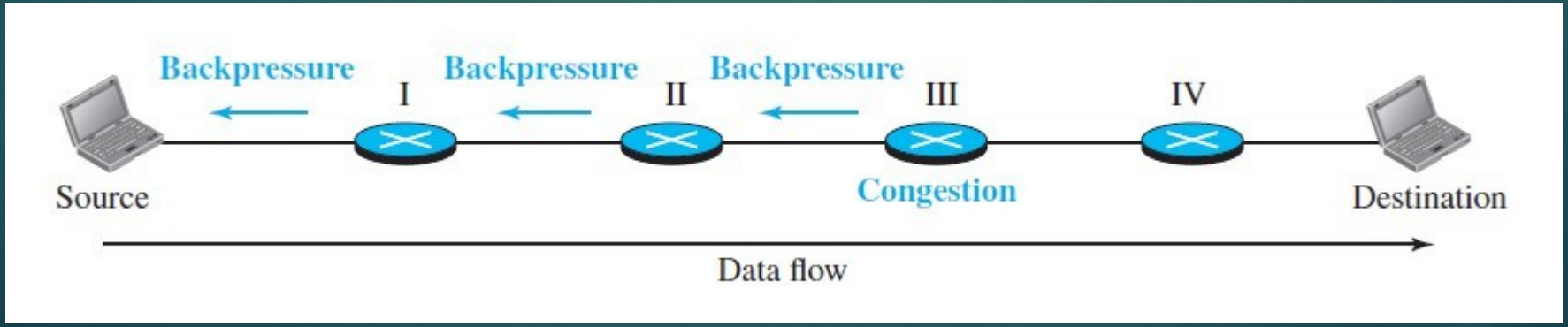
- Closed-loop congestion control mechanisms try to alleviate congestion after it happens. Several mechanisms have been used by different protocols
- Backpressure
- Choke Packet
- Implicit Signaling
- Explicit Signaling

Prepared By: Er. Sushant Bhattarai

Closed Loop Congestion Control

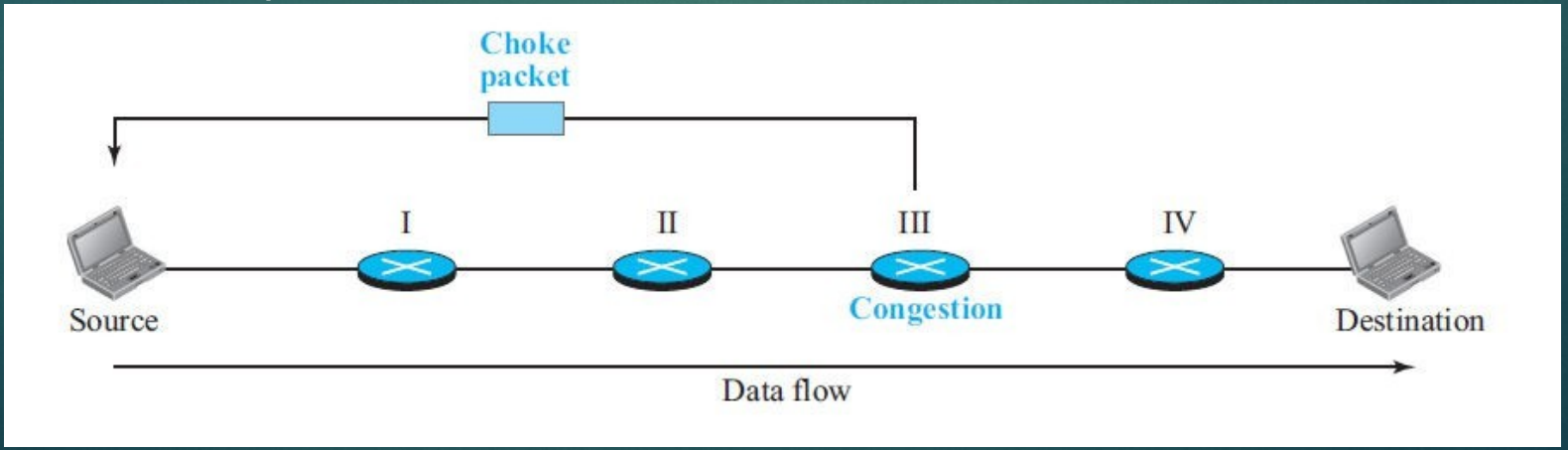
Prepared By: Er. Sushant Bhattarai

- Backpressure
  - The technique of backpressure refers to a congestion control mechanism in which a congested node stops receiving data from the immediate upstream node or nodes.
  - This may cause the upstream node or nodes to become congested, and they, in turn, reject data from their upstream node or nodes, and so on. Backpressure is a node-to- node congestion control that starts with a node and propagates, in the opposite direction of data flow, to the source.
  - The backpressure technique can be applied only to virtual circuit networks, in which each node knows the upstream node from which a flow of data is coming.



Closed Loop Congestion Control

- Choke Packet
  - A **choke packet** is a packet sent by a node to the source to inform it of congestion.
  - In the choke-packet method, the warning is from the router, which has encountered congestion, directly to the source station. The intermediate nodes through which the packet has traveled are not warned.
  - When a router in the Internet is overwhelmed with IP datagrams, it may discard some of them, but it informs the source host, using a source quench ICMP message.
  - The warning message goes directly to the source station; the intermediate routers do not take any action



Prepared By: Er. Sushant Bhattarai

Closed Loop Congestion Control

- Implicit Signaling
  - In implicit signaling, there is no communication between the congested node or nodes and the source. The source guesses that there is congestion somewhere in the network from other symptoms.
  - For example, when a source sends several packets and there is no acknowledgment for a while, one assumption is that the network is congested. The delay in receiving an acknowledgment is interpreted as congestion in the network; the source should slow down.

Prepared By: Er. Sushant Bhattarai

Closed Loop Congestion Control

- Explicit Signaling
  - The node that experiences congestion can explicitly send a signal to the source or destination.
  - The explicit-signaling method, however, is different from the choke-packet method. In the choke-packet method, a separate packet is used for this purpose; in the explicit-signaling method, the signal is included in the packets that carry data.
  - Explicit signaling can occur in either the forward or the backward direction

Traffic Shaping Algorithm**(VVI)**

- To control the amount and the rate of traffic is called traffic shaping or traffic policing.
- The first term is used when the traffic leaves a network; the second term is used when the data enters the network.
- Two techniques can shape or police the traffic:
- Leaky bucket algorithm
- Token bucket algorithm

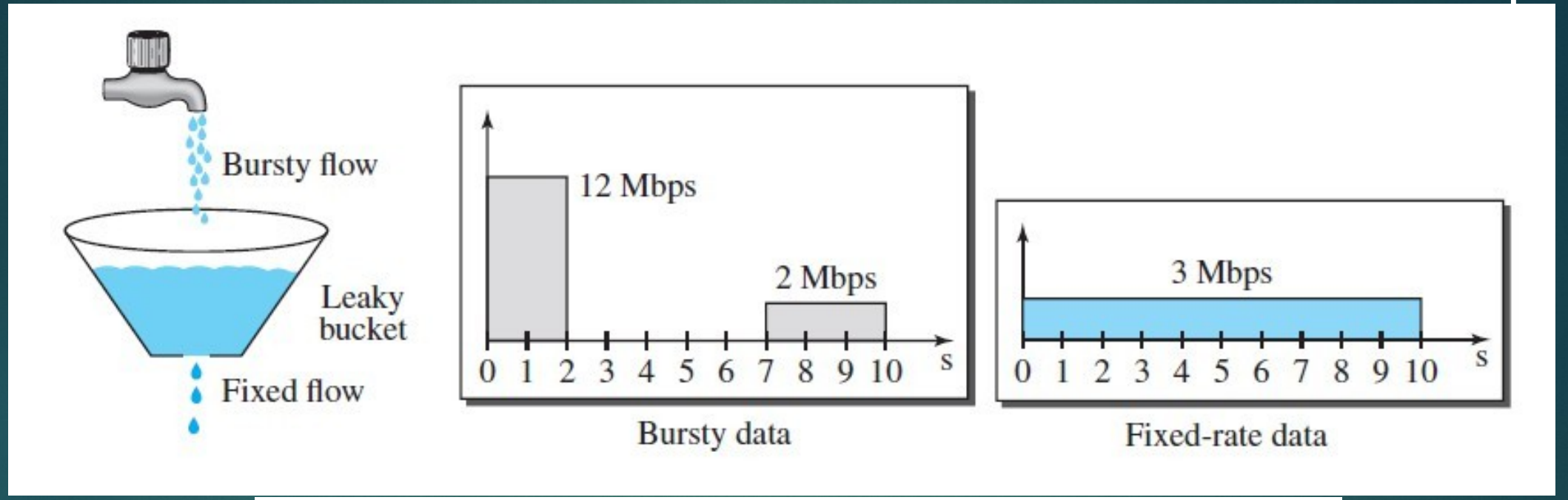
Prepared By: Er. Sushant Bhattarai

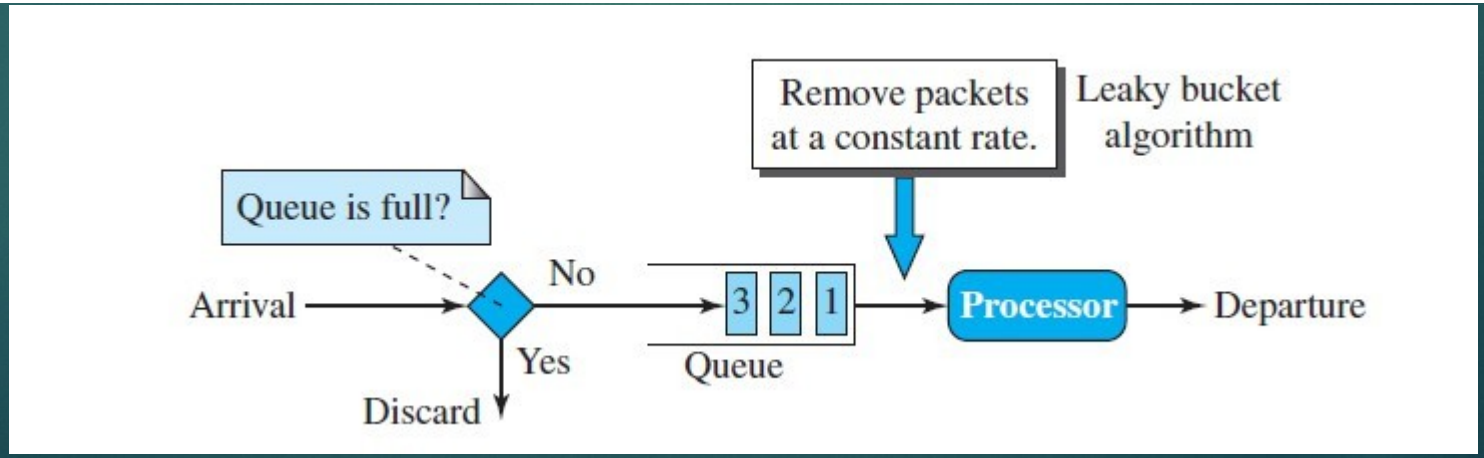
Leaky Bucket Algorithm

Prepared By: Er. Sushant Bhattarai

- If a bucket has a small hole at the bottom, the water leaks from the bucket at a constant rate as long as there is water in the bucket.
- The rate at which the water leaks does not depend on the rate at which the water is input unless the bucket is empty. If the bucket is full, the water overflows.
- The input rate can vary, but the output rate remains constant. Similarly, in networking, a technique called leaky bucket can smooth out bursty traffic.
- Bursty chunks are stored in the bucket and sent out at an average rate.
- Similarly, each network interface contains a leaky bucket and the following steps are involved in leaky bucket algorithm:
- When host wants to send packet, packet is thrown into the bucket.
- The bucket leaks at a constant rate, meaning the network interface transmits packets at a constant rate.
- Busty traffic is converted to a uniform traffic by the leaky bucket.
- In practice the bucket is a finite queue that outputs at a finite rate.

Leaky Bucket Algorithm



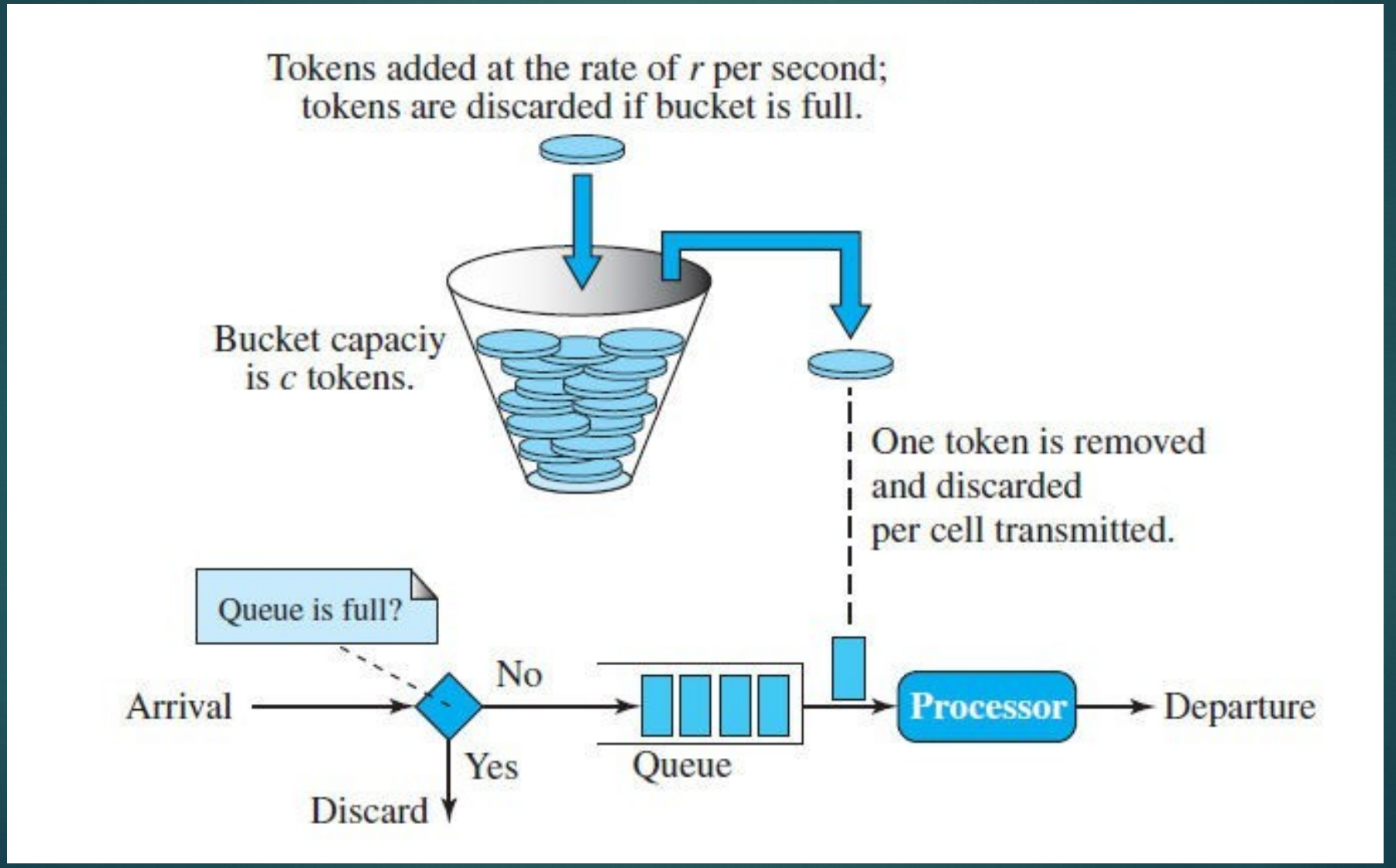


Token Bucket Algorithm

- The leaky bucket algorithm allows only an average (constant) rate of data flow.Its major problem is that it cannot deal with busty data.
- A leaky bucket algorithm does not consider the idle time of the host
- To overcome this problem, a token bucket algorithm is used. A token bucket algorithm allows busty data transfers.
- A token bucket algorithm is a modification of leaky bucket in which leaky bucket contains tokens.
- In this algorithm, a token(s) are generated at every clock tick. For a packet to be transmitted, system must remove token(s) from the bucket.
- Thus, a token bucket algorithm allows idle hosts to accumulate credit for the future in form of tokens.
- For example, if a system generates 100 tokens in one clock tick and the host is idle for 100 ticks. The bucket will contain 10,000 tokens.
- Now, if the host wants to send busty data, it can consume all 10,000 tokens at once for sending 10,000 cells or bytes.
- Thus a host can send busty data as long as bucket is not empty.

Prepared By: Er. Sushant Bhattarai

Token Bucket Algorithm



Prepared By: Er. Sushant Bhattarai

Techniques to improve QOS

- **Quality of service** (**QoS**) is an internetworking issue that refers to a set of techniques and mechanisms that guarantee the performance of the network to deliver predictable service to an application program.
- Although formal classes of flow are not defined in the Internet, an IP datagram has a ToS(Type of Service) field that can informally define the type of service required for a set of datagrams sent by an application
- Scheduling
- Traffic Shaping
- Resource Reservation
- Admission Control

Techniques to improve QOS

- Scheduling
  - Treating packets (datagrams) in the Internet based on their required level of service can mostly happen at the routers. It is at a router that a packet may be delayed, suffer from jitters, be lost, or be assigned the required bandwidth.
  - A good scheduling technique treats the different flows in a fair and appropriate manner. Several scheduling techniques are designed to improve the quality of service
- Traffic Shaping
  - To control the amount and the rate of traffic is called traffic shaping or traffic policing.
  - The first term is used when the traffic leaves a network; the second term is used when the data enters the network. Two techniques can shape or police the traffic: leaky bucket and token bucket.

Techniques to improve QOS

- Resource Reservation
  - A flow of data needs resources such as a buffer, bandwidth, CPU time, and so on. The quality of service is improved if these resources are reserved beforehand
- Admission Control
  - Admission control refers to the mechanism used by a router or a switch to accept or reject a flow based on predefined parameters called flow specifications.
  - Before a router accepts a flow for processing, it checks the flow specifications to see if its capacity can handle the new flow. It takes into account bandwidth, buffer size, CPU speed, etc., as well as its previous commitments to other flows.

Prepared By: Er. Sushant Bhattarai

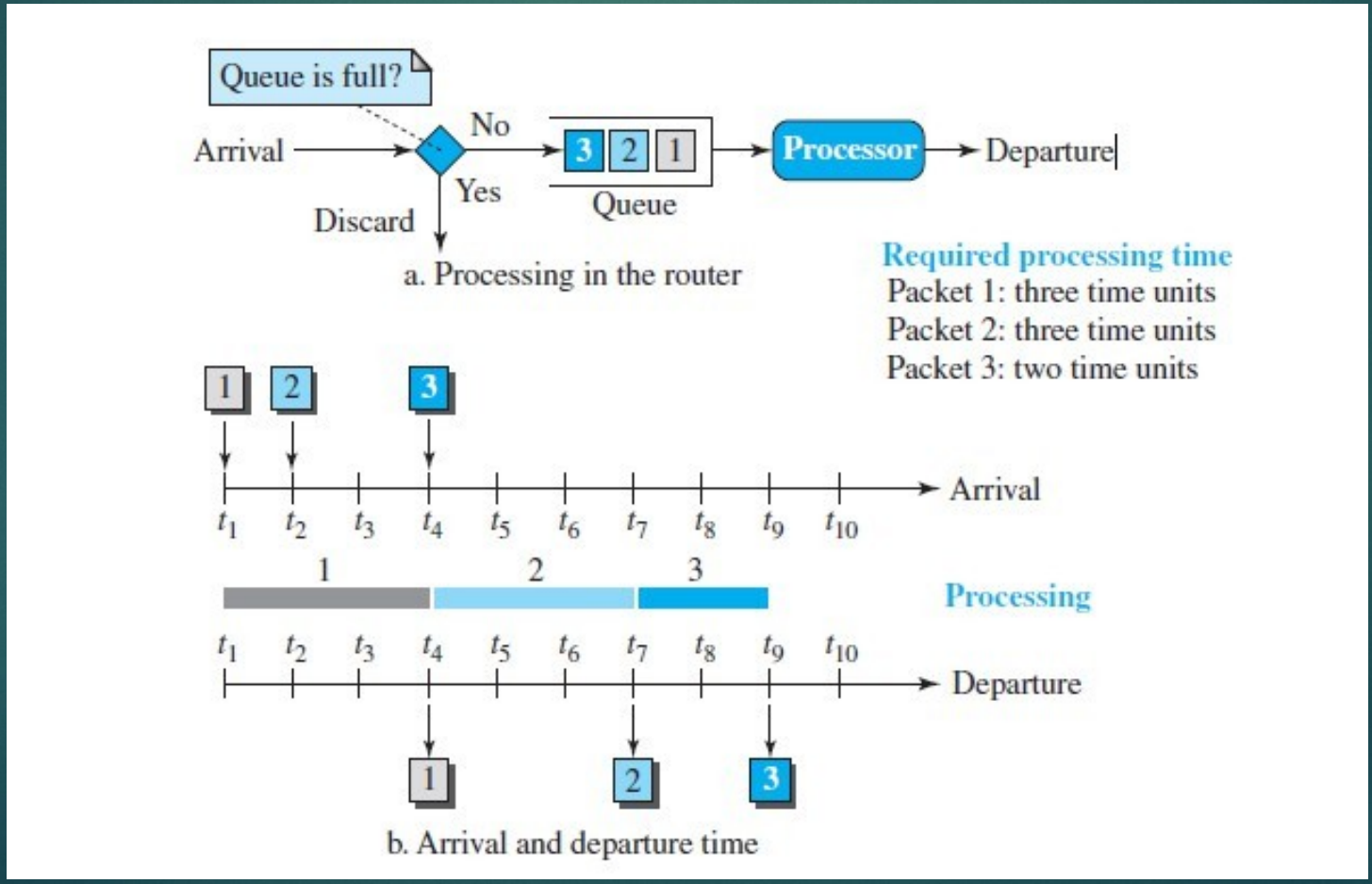
Queuing Technique for Scheduling

- Three major technique for scheduling
- FIFO queuing
- Priority queuing
- Weighted fair queuing.

Prepared By: Er. Sushant Bhattarai

FIFO queuing

- In **first-in, first-out (FIFO) queuing,** packets wait in a buffer (queue) until the node (router) is ready to process them.
- If the average arrival rate is higher than the average processing rate, the queue will fill up and new packets will be discarded.
- A FIFO queue is familiar to those who have had to wait for a bus at a bus stop.



Prepared By: Er. Sushant Bhattarai

FIFO queuing

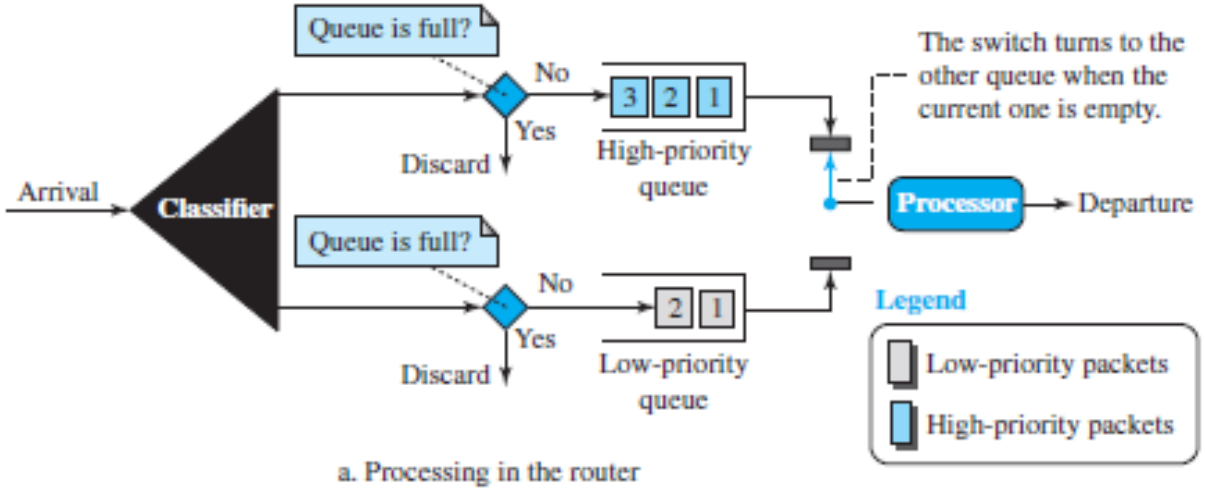
- Figure below shows a conceptual view of a FIFO queue. The figure also shows the timing relationship between arrival and departure of packets in this queuing.
- Packets from different applications (with different sizes) arrive at the queue,
are processed, and depart.
- A larger packet definitely may need a longer processing time. In the figure, packets 1 and 2 need three time units of processing, but packet 3, which is smaller, needs two time units.
- This means that packets may arrive with some delays but depart with different delays.

Prepared By: Er. Sushant Bhattarai

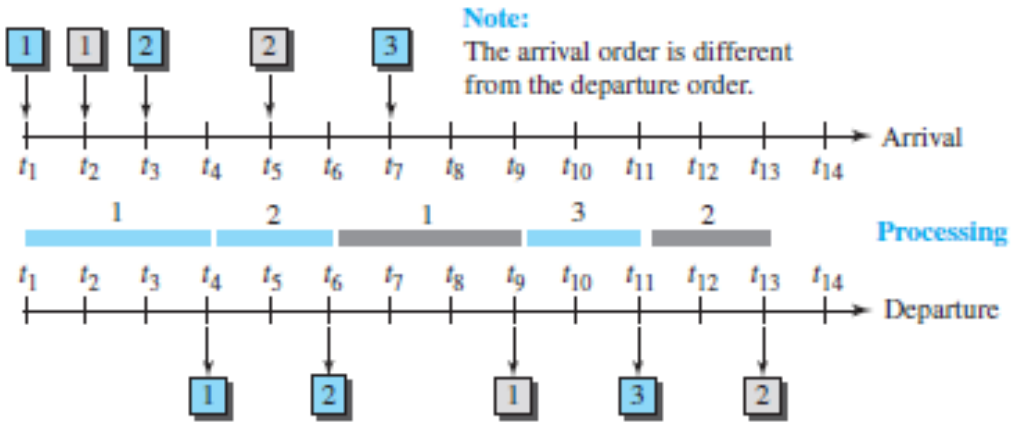
Priority Queuing

- Queuing delay in FIFO queuing often degrades quality of service in the network. A frame carrying real-time packets may have to wait a long time behind a frame carrying a small file.
- We solve this problem using multiple queues and priority queuing. In **priority queuing,** packets are first assigned to a priority class.
- Each priority class has its own queue.
- The packets in the highest-priority queue are processed first. Packets in the lowest-priority queue are processed last.
- The system does not stop serving a queue until it is empty.
- A packet priority is determined from a specific field in the packet header: the ToS field of an IPv4 header, the priority field of IPv6, a priority number assigned to a destination address, or a priority number assigned to an application (destination port number), and so on.

Priority Queuing



a. Processing in the router



b. Arrival and departure time

Prepared By: Er. Sushant Bhattarai

Priority Queuing

- ► Figure above showed priority queuing with two priority levels (for simplicity). A priority queue can provide better QoS than the FIFO queue because higher-priority traffic, such as multimedia, can reach the destination with less delay.
- ► If there is a continuous flow in a high-priority queue, the packets in the lower-priority queues will never have a chance to be processed. This is a condition called *starvation*.
- ► Severe starvation may result in dropping of some packets of lower priority. In the figure, the packets of higher priority are sent out before the packets of lower priority.

Weighted Fair Queuing

- A better scheduling method is **weighted fair queuing.** In this technique, the packets are still assigned to different classes and admitted to different queues.
- The queues, however, are weighted based on the priority of the queues; higher priority means a higher weight.
- The system processes packets in each queue in a round-robin fashion with the number of packets selected from each queue based on the corresponding weight.
- For example, if the weights are 3, 2, and 1, three packets are processed from the first queue, two from the second queue, and one from the third queue. In this way, we have fair queuing with priority.

Weighted Fair Queuing